APPARATUS, METHOD, SYSTEM AND EXECUTABLE MODULE FOR CONFIGURATION AND OPERATION OF ADAPTIVE INTEGRATED CIRCUITRY HAVING FIXED, APPLICATION SPECIFIC COMPUTATIONAL ELEMENTS

Field of the Invention

The present invention relates, in general, to integrated circuits and systems of integrated circuits. More particularly, the present invention relates to an apparatus, method, system and executable module for configuration and operation of adaptive integrated circuitry having fixed, application specific computational elements.

Cross-Reference to Related Applications

This application is related to Paul L. Master et al., U. S. Patent Application Serial No. 09/815,122, entitled "Adaptive Integrated Circuitry With Heterogeneous And Reconfigurable Matrices Of Diverse And Adaptive Computational Units Having Fixed, Application Specific Computational Elements", filed March 22, 2001, commonly assigned to QuickSilver Technology, Inc., and incorporated by reference herein, with priority claimed for all commonly disclosed subject matter (the "first related application").

~~This application is related to Paul L. Master et al., U. S. Patent Application Serial No. ____, entitled "Apparatus, System And Method For Configuration Of Adaptive Integrated Circuitry Having Fixed, Application Specific Computational Elements", filed concurrently herewith, commonly assigned to QuickSilver Technology, Inc., and incorporated by reference herein, with priority claimed for all commonly disclosed subject matter (the "second related application").~~

Background of the Invention

The first related application discloses a new form or type of integrated circuitry which effectively and efficiently combines and maximizes the various

advantages of processors, application specific integrated circuits ("ASICs"), and field programmable gate arrays ("FPGAs"), while minimizing potential disadvantages. The first related application illustrates a new form or type of integrated circuit, referred to as an adaptive computing engine ("ACE"), which provides the programming flexibility of a processor, the post-fabrication flexibility of FPGAs, and the high speed and high utilization factors of an ASIC. This ACE integrated circuitry is readily reconfigurable, is capable of having corresponding, multiple modes of operation, and further minimizes power consumption while increasing performance, with particular suitability for low power applications, such as for use in hand-held and other battery-powered devices.

The second related application discloses a preferred system embodiment that includes an ACE integrated circuit coupled with one or more sets of configuration information. This configuration information is required to generate, in advance or in real-time (or potentially at a slower rate), the configurations and reconfigurations which provide and create one or more operating modes for the ACE circuit, such as wireless communication, radio reception, personal digital assistance ("PDA"), MP3 or MP4 music playing, or any other desired functions. Various methods, apparatuses and systems are also illustrated in the second related application for generating and providing configuration information for an ACE integrated circuit, for determining ACE reconfiguration capacity or capability, for providing secure and authorized configurations, and for providing appropriate monitoring of configuration and content usage.

A need remains, however, for an apparatus, method and system for not only configuring, but also operating such adaptive integrated circuitry, with one or more operating modes or other functionality of ACE circuitry and other ACE devices. Such an apparatus, method and system should be capable of configuring and operating the adaptive IC, utilizing both configuration information provided independently of user data or other content, and utilizing configuration information provided concurrently with user data or other content. Such an apparatus, method and system should provide the means to, among other things, coordinate configuration with data, provide self-routing of configuration and data, and provide power control within ACE circuitry.

Summary of the Invention

The adaptive computing engine ("ACE") circuit of the present invention, for adaptive or reconfigurable computing, includes a plurality of heterogeneous computational elements coupled to an interconnection network (rather than the same, homogeneous repeating and arrayed units of FPGAs). The plurality of heterogeneous computational elements include corresponding computational elements having fixed and differing architectures, such as fixed architectures for different functions such as memory, addition, multiplication, complex multiplication, subtraction, configuration, reconfiguration, control, input, output, routing, and field programmability.

In response to configuration information, the interconnection network is operative, in advance, in real-time or potentially slower, to configure and reconfigure the plurality of heterogeneous computational elements for a plurality of different functional modes, including linear algorithmic operations, non-linear algorithmic operations, finite state machine operations, memory operations, and bit-level manipulations. In turn, this configuration and reconfiguration of heterogeneous computational elements, forming various computational units and adaptive matrices, generates the selected, higher-level operating mode of the ACE integrated circuit, for the performance of a wide variety of tasks.

The present invention illustrates various means for both configuring and operating such adaptive integrated circuitry, for one or more operating modes or other functionality of ACE circuitry and other ACE devices. The present invention provides such configuration and operation of the adaptive IC, utilizing both configuration information provided independently of user data or other content, and utilizing configuration information provided concurrently with user data or other content. The present invention also provides the means to, among other things, coordinate configuration with data, provide self-routing of configuration and data, and provide power control within ACE circuitry.

A preferred method of providing such configuration and operation utilizes a "silverware" module (also referred to as "silverware") comprised of a plurality of information sequences. A first information sequence (or field) provides configuration control, which may be either configuration information or a reference (such as a flag or

other designation) to corresponding configuration information cached or stored in memory (or stored in a configuration of computational elements). A second information sequence provides operand data for use by configured computational elements. A third information sequence provides routing control, to direct the other information sequences to their appropriate locations within the matrix environment of the ACE integrated circuitry. Also in the preferred embodiment a fourth information sequence is utilized to provide power control, to clock on or off various computational elements. Other information sequences may also be utilized, for example, to maintain configuration instantiations for repeated use, or to define new fields or types of information for future use (which are currently undefined).

For example, one of the preferred system embodiments provides, first, means for routing configuration information to a plurality of computational elements; second, means for configuring and reconfiguring a plurality of computational elements to form a plurality of configured computational elements for the performance of a plurality of selected functions; third, means for providing operand data to the plurality of configured computational elements; and fourth, means for controlling configuration timing to precede a receipt of corresponding operand data.

Another preferred system embodiment provides, first, means for spatially configuring and reconfiguring a plurality of computational elements to form a first plurality of configured computational elements for the performance of a first plurality of selected functions; second, means for temporally configuring the plurality of computational elements to form a second plurality of configured computational elements for the performance of a second plurality of selected functions; third, means for providing data to the first and second pluralities of configured computational elements; and fourth, means for coordinating the spatial and temporal configurations of the plurality of computational elements with the provision of the data to the first and second pluralities of configured computational elements.

Numerous other advantages and features of the present invention will become readily apparent from the following detailed description of the invention and the embodiments thereof, from the claims and from the accompanying drawings.

Brief Description of the Drawings

Figure 1 is a diagram illustrating an exemplary executable information module in accordance with the present invention.

Figure 2 is a block diagram illustrating a plurality of system embodiments in accordance with the present invention.

Figure 3 is a block diagram illustrating an integrated system embodiment in accordance with the present invention.

Figure 4 is a block diagram illustrating a preferred adaptive computing engine (ACE) embodiment in accordance with the present invention.

Figure 5 is a block diagram illustrating a reconfigurable matrix, a plurality of computation units, and a plurality of computational elements, in accordance with the present invention.

Figure 6 is a block diagram illustrating, in greater detail, a computational unit of a reconfigurable matrix in accordance with the present invention.

Figure 7 is a block diagram illustrating, in detail, a preferred multi-function adaptive computational unit having a plurality of different, fixed computational elements, in accordance with the present invention.

Figure 8 is a block diagram illustrating, in detail, a preferred adaptive logic processor computational unit having a plurality of fixed computational elements, in accordance with the present invention.

Figure 9 is a block diagram illustrating, in greater detail, a preferred core cell of an adaptive logic processor computational unit with a fixed computational element, in accordance with the present invention.

Figure 10 is a block diagram illustrating, in greater detail, a preferred fixed computational element of a core cell of an adaptive logic processor computational unit, in accordance with the present invention.

Detailed Description of the Invention

While the present invention is susceptible of embodiment in many different forms, there are shown in the drawings and will be described herein in detail specific embodiments thereof, with the understanding that the present disclosure is to be

considered as an exemplification of the principles of the invention and is not intended to limit the invention to the specific embodiments illustrated.

As indicated above, a need remains for an apparatus, method and system for configuring and operating adaptive integrated circuitry, to provide one or more operating modes of ACE circuitry and other devices incorporating ACE technology. Such an apparatus, method and system are provided in accordance with the present invention, and are capable of configuring and operating the adaptive IC, utilizing both configuration information provided independently of user data or other content, and utilizing configuration information provided concurrently with user data or other content. The present invention also provides the means to, among other things, coordinate configuration with data, provide self-routing of configuration and data, and provide power control within ACE circuitry.

The apparatus, systems and methods of the present invention utilize a new form of integrated circuitry, referred to as an adaptive computing engine. The ACE architecture utilizes a plurality of fixed computational elements, such as correlators, multipliers, complex multipliers, adders, routers, demodulators, and combiners, which may be configured and reconfigured, in advance, in real-time or potentially at a slower rate, through an interconnection network, in response to configuration information, to form the functional blocks (computational units and matrices) which may be needed, at any given time, to execute or perform the selected operating mode, such as to perform wireless communication functionality. The methodology and systems of the present invention also minimize power consumption and are especially suitable for low power applications, such as for use in hand-held and other battery-powered devices.

Figure 1 is a diagram illustrating an exemplary executable information module 70, preferably referred to as a "silverware module", in accordance with the present invention. The module 70 may be implemented as one or more discrete information packets, such as internet protocol (IP) packets, or may be implemented as a continuous stream of information or other bit stream, as discussed in greater detail below.

Referring to Figure 1, the module 70 consists of a plurality of information fields, some of which are requisite and some of which are optional. In addition, depending upon the chosen embodiment, the various fields (71 – 99) may occur in a

plurality of different orders, and in some embodiments, without regard to order. As illustrated in Figure 1, the module 70 includes header information in field 71, such as synchronization, addressing, and security information (such as digital signatures). Such header information is typically included when the module 70 is transmitted or transferred to an ACE circuit from an external source, such as those illustrated in Figure 2.

Next, fields 72 – 78 illustrate configuration information with corresponding self-routing information. As discussed in greater detail below, this routing information has two purposes: first, it directs the configuration information to a cache or memory location for storage within the various matrices of the ACE architecture, and second, it directs the configuration information to its designated or specified location to configure computational elements within the various matrices of the ACE architecture. (It should be noted that once configured, the computational elements and interconnection network effectively also operate as a memory, storing the configuration information as the actual configuration.) The routing information may be provided to the ACE by an external source or may be self-generated by the ACE architecture. As illustrated in Figure 1, fields 72 and 73 provide routing information for configuration "A" and configuration information for configuration "A", respectively; fields 74 and 75 provide routing information for configuration "B" and configuration information for configuration "B", respectively; fields 76 and 77 provide routing information for configuration "C" and configuration information for configuration "C", respectively; and fields 78 and 79 provide routing information for configuration "D" and configuration information for configuration "D", respectively.

Such routing and configuration information, in the preferred embodiment, are provided for all configurations to be utilized in providing one or more operating modes for ACE circuits and devices. As illustrated below, there are many instances in which only configuration information is provided to an ACE device, which may then internally generate its own routing information. In other cases, both types of information may be provided to an ACE from an external source. Following such configuration, for example, as a mobile communication device, user data may be provided separately, such as voice data during a mobile communication session.

In yet other cases, such configuration and routing information may be provided concurrently with user data. For example, an MPEG file may be downloaded to an ACE device, consisting of both configuration information and the music content to be played. For these circumstances, and for the internal operation of the ACE architecture as discussed in greater detail below, additional information is included in the module 70. Referring to Figure 1, the module 70 preferably includes references or flags to indicate previously provided and stored configuration information, such as field 80, providing a reference or flag to configuration information "A" and field 81, providing a reference or flag to configuration information "B". These references or flags are used to coordinate the timing of configurations with respect to arriving data, *i.e.,* to "call", initiate or otherwise direct the occurrence of these configurations prior to a receipt of data by these configured computational elements. Next, as illustrated, the module 70 (optionally) includes a power control field 82, which is utilized to separately and independently clock (or power) the various components of the ACE architecture, for example, to provide clocking to configurations "A" and "B", while saving power in then currently unused portions of the IC.

Continuing to refer to Figure 1, fields 83 – 97, among other things, illustrate the provision of user data to the ACE architecture, namely, the data to be utilized, operated upon or "crunched" by the various configured computational elements in performing their various functions ("user data" or "operand data"), such as discrete cosine transformation, and may also include other data or parameters useful in establishing or restoring settings of the various computational elements, such as previously derived equalizer coefficients ("coefficient data"). (Such operand or user data, as used to herein, provides a "shorthand" distinction among types of information, distinguishing data to be "crunched" from configuration information, configuration data, or other types of information, such as routing and clocking information.) Routing information is also utilized to provide self-routing of the data to their appropriate matrix locations. In addition, an optional field may be included to designate types of information, such as configuration information or data information.

As illustrated, fields 83 and 84 provide routing information for the data for configuration "A" and the data to be used in or by configuration "A", respectively; field

85 provides a reference or flag to generate configuration "C"; fields 86 and 87 provide routing information for the data for configuration "B" and the data to be used in or by configuration "B", respectively; field 88 provides a second or substitute routing location for configuration "D" (such as a different location within the various matrices), and field 89 provides a reference or flag to generate configuration "D"; fields 90 and 91 provide routing information for the data for configuration "A" and additional data to be used in or by configuration "A", respectively; fields 92 and 93 provide routing information for the data for configuration "C" and the data to be used in or by configuration "C", respectively; fields 94 and 95 provide routing information for the data for configuration "D" and the data to be used in or by configuration "D", respectively; and fields 96 and 97 provide routing information for the data for configuration "B" and the data to be used in or by configuration "B", respectively. Another field (98) may be used to provide information concerning or designating an information type (for example, that configuration information will be the next fields in the module 70). As another option, an additional field (field 99) may also be utilized for "loop" instructions, to indicate that a particular instantiation of a configuration is to remain in place for a particular duration or number of cycles. Other fields may also be utilized, for example, to define new types of information for future use (which are currently undefined), or otherwise to be self-extensible. As illustrated, the module 70 may continue, providing more configuration information and data (with corresponding routing information, power control, type designations, and so on), for as long as the ACE architecture is being utilized or operated. The use of the information provided in module 70 is also discussed in greater detail below.

Figure 2 is a block diagram illustrating a plurality of system embodiments in accordance with the present invention. As indicated above, and as discussed in greater detail below, the preferred system of the present invention consists of an ACE 100 coupled or combined with configuration information (such as a module 70), and may be implemented in a wide variety of embodiments including, for example, within wireless devices 30 and 32, wireline device 35, computers 55, consumer electronics, automobile electronics 37, and network infrastructure equipment, such as servers 54, routers 53, local area network (LAN) 41, wireless LAN 43, wide area network (WAN) 42, adjunct

network entity 50, switching systems 52 and 56, wireless base stations 25, and any other electronic device.

As a point of clarification, the terminology "configuration information", as used herein, should be understood generally to have and include its linguistic, plural connotation, *i.e.,* configuration information is a plurality of information bits, groups or sets of information, namely, a "plurality" of configuration information. For example, "configuration information" may be viewed as being a set of configuration information comprised of a plurality of subsets, such subsets being first configuration information, second configuration information, third configuration information, and so on, through n^{th} configuration information. Although a subset of configuration information may be singular (one bit of information contained in the subset), each such subset of configuration information is also generally plural, typically including more information than may be encoded by a single bit, such as 8, 16, 32 or 64 information bits.

Configuration information, such as that illustrated in module 70, with or without user or coefficient data, may also exist in a variety of forms, and at any given time, may have a stored (or fixed) nature, or may have a transient or temporal nature. For example, as illustrated in Figure 2, executable modules (such as module 70), or other configuration information (without the form of module 70), may be stored as a binary (bit) file in a flash memory 10 (for device 35) or in a computer or other machine-readable medium 20 (such as a CD-ROM, other optical drive, computer memory, hard drive or floppy disk) for computer 55B. As discussed in greater detail below, such configuration information may also be interdigitated or intertwined with data, forming a silverware module such as module 70, and also stored as a binary (bit) file in a silverware storage media 15 or other medium (such as flash memory or CD-ROM). The module 70 or configuration information may also occur transiently and across time, for example, when wirelessly downloaded from a base station 25A to a wireless device 32 (such as a mobile station or other mobile telephone) over an air interface.

Referring to Figure 2 in greater detail, a plurality of networks are illustrated, including local area network ("LAN") 41, wireless LAN 43, wide area network ("WAN") 42, and, more generally, network 40, such as a public switched telephone network ("PSTN") or internet. Coupled to the various networks are routers

53A and 53B, servers 54A and 54B, wireline switching center 56, mobile switching center ("MSC") 52, with further connection or couplability to wireless base stations (or other wireless transceivers) 25A and 25B, wireline device 35, computers 55A and 55B, and adjunct network entity 50. As known in the art, these various devices may be connected via trunking, optical and other signaling lines to each other and to broader networks (such as to a PSTN or internet), with multiple communication connections to other locations, such as providing a link to a satellite (not separately illustrated) and providing other wireless links (air interfaces). Router 53B, server 54B, base station 25B, and computer 55B are separately designated (with "B") to illustrate the potential inclusion of an ACE 100 (and the systems of the present invention) within such infrastructure equipment, and within local area network (LAN) 41, wireless LAN 43, wide area network (WAN) 42, adjunct network entity 50, in addition to inclusion within consumer, automotive, and mobile electronics. Also, while the wireline and mobile switching centers 56 and 52 are usually physically separated due to regulatory and other historical or legacy reasons, these switching centers may also be combined into one or more switching centers having both wireline and wireless functionalities.

These various server, switching, routing and other entities may also be connected through network 40 to one or more intelligent network devices referred to as an adjunct network entities, such as adjunct network entity 50, which may be an additional type of server, database, a service control point ("SCP"), a service circuit node ("SCN") (also referred to as a service node), an intelligent peripheral ("IP"), a gateway, or another intelligent network device. One or more adjunct network entities 50 are preferably connected or coupled to a network 40, for direct or indirect connection to wireline switching center 56, MSC 52, local area network (LAN) 41, wireless LAN 43, wide area network (WAN) 42, routers 53 and servers 54. In the preferred embodiment, an adjunct network entity 50 provides a node or platform for particular applications ("application nodes") 51, illustrated as application nodes 51A, 51B through 51N, to perform various functions such as providing downloads of configuration information, executable modules 70, authentication, security, authorization, and compatibility evaluation. In addition to inclusion within an adjunct network entity 50, these various application nodes 51 may also be distributed among or included within the other various

devices, such as within one or more servers 54. For example, one server 54 may be utilized to provide configuration information, with an adjunct network entity 50 utilized for authentication and security, with tracking and accounting occurring at yet another server 54 or computer 55.

~~For purposes of explanation and not limitation, the various systems of the~~ present invention, as illustrated in Figure 2, include: system 11 (ACE 100 of wireline device 35 with configuration information or modules 70 in FLASH 10); system 16 (ACE 100 of wireless device 30 with configuration information or modules 70 in silverware storage medium 15); system 31 (ACE 100 of wireless device 32 with configuration information or modules 70 stored in a form of memory (separately illustrated in Figure 3), such as RAM or a matrix interconnection network ("MIN"), discussed below; system 21 (ACE 100 of computer 55B with configuration information or modules 70 stored in computer readable medium 20; system 22 (ACE 100 of server 54B with configuration information or modules 70 stored in a form of memory (separately illustrated in Figure 3); and system 23 (ACE 100 of router 53B with configuration information or modules 70 stored in a memory (separately illustrated in Figure 3). As may be apparent, a system of the present invention may be embodied within any device or other article, in addition to those illustrated (*e.g.*, LAN 41, wireless LAN 43, WAN 42, and adjunct network entity 50), which include both an ACE 100 and configuration information (or module 70) for the provision of a corresponding operating mode, and may otherwise be co-extensive ~~with any particular apparatus or other embodiment.~~

Other network or distribution level systems are also included within the scope of the present invention. Exemplary network systems may include one or more application nodes 51, in an adjunct network entity 50 or other server 54, which provide configuration information or silverware modules (configuration information coupled with data), such as a module 70, for use by an ACE 100. By storing such configuration and other information, such network or distribution level systems effectively store "hardware" on the "net". Such network or distribution level systems, in response to a request from or on behalf of an ACE 100, in the preferred embodiment, may provide one or more of the following: one or more sets of configuration information; content or other data modified for use with configuration information; silverware modules (70) combining

configuration information with corresponding data or other content; configuration information tailored or watermarked for a unique device; and/or encryption of configuration information or silverware modules.

Distributed systems are also within the scope of the present invention, as configuration information does not need to be local to any given ACE 100 device. For example, configuration information or silverware may be stored across a network 40, such as between and among application nodes 51, adjunct network entity 50, other server 54, and the other illustrated elements of Figure 1. For such distributed systems, the ACE 100 may only be configured, such as through an operating system ("OS"), to obtain the configuration information, such as through one of these network devices.

Other distributed systems, within the scope of the present invention, are comprised of clusters of ACE 100 devices, which are configured to be aware of each other. For example, wireless IP routing could occur by nearest neighboring ACEs, each configured for both reception and transmission operating modes. Other ACE clusters could perform parallel processing tasks, act as a distributed antenna system, or otherwise perform interactive functions.

Figure 3 is a block diagram illustrating an integrated system embodiment 60 in accordance with the present invention. The system 60 is preferably implemented as a single integrated circuit (system on a chip or "SOC"). The system 60 includes an ACE 100, and may also include a memory 61, an interface 62 and one or more other processing elements 65. Such a system 60, for example, may be included within routers 53 and servers 54 of Figure 2, or may be included within other embedded systems, such as within mobile stations or devices 30 and 32, wireline device 35, and so on. When the system 60 is comprised solely of an ACE 100, as discussed in greater detail below, that ACE 100 will generally be configured to include processing, interface and other I/O functionality, with memory configured either through memory computational elements or directly within the matrix interconnection network (MIN). The system 60, as illustrated in Figure 2 with optional processing element 65, interface 62, and memory 61, will typically be implemented to provide backwards or retro-compatibility with existing or other legacy systems and devices.

Sub A3

~~The interface 62 is utilized for appropriate connection to a relevant~~ channel, network or bus; for example, the interface 62 may provide impedance matching, drivers and other functions for a wireline interface, may provide demodulation and analog to digital conversion for a wireless interface, and may provide a physical interface for the memory 61 with other devices. In general, the interface 62 is used to receive and transmit data, depending upon the selected embodiment, such as voice information, configuration information, silverware modules (70), control messages, authentication data and other pertinent information. The ACE 100 may also be configured to provide the functionality of the interface 62, including internal IC I/O and external (off-chip) I/O, such as for PCI bus control. The memory 61 may be an integrated circuit or portion of an integrated circuit, such as various forms of RAM, DRAM, SRAM, FeRAM, MRAM, ROM, EPROM, E^2PROM, flash, and so on. For non-IC (or non-SOC) embodiments, the memory 61 may also be a magnetic (hard or floppy) drive, an optical storage device, or any other type of data storage apparatus and, as indicated above, may be distributed across multiple devices. In addition, depending upon the selected embodiment, and as discussed in greater detail below, the memory 61 may also be included within the ACE 100, through memory computational elements or within the matrix interconnection network (MIN). One or more processing elements 65 optionally may be included within system 60, to provide any additional processing capability, such as reduced instruction set ~~("RISC") processing, or may be included as computational elements within the ACE 100.~~

The use and/or creation of modules 70, and the operation of the various systems illustrated in Figures 2 and 3 are discussed in greater detail below, with reference to Figures 4 – 10 and corresponding explanation of the ACE 100 architecture.

Figure 4 is a block diagram illustrating a preferred ACE apparatus 100 embodiment in accordance with the present invention. The ACE 100 is preferably embodied as an integrated circuit, or as a portion of an integrated circuit having other, additional components. (The ACE 100 is also described in detail in the related application.) In the preferred embodiment, and as discussed in greater detail below, the ACE 100 includes one or more reconfigurable matrices (or nodes) 150, such as matrices 150A through 150N as illustrated, and a matrix interconnection network (MIN) 110. Also in the preferred embodiment, and as discussed in detail below, one or more of the

matrices 150, such as matrices 150A and 150B, are configured for functionality as a controller 120, while other matrices, such as matrices 150C and 150D, are configured for functionality as a memory 140. While illustrated as separate matrices 150A through 150D, it should be noted that these control and memory functionalities may be, and preferably are, distributed across a plurality of matrices 150 having additional functions to, for example, avoid any processing or memory "bottlenecks" or other limitations. Such distributed functionality, for example, is illustrated in Figure 5. The various matrices 150 and matrix interconnection network 110 may also be implemented together as fractal subunits, which may be scaled from a few nodes to thousands of nodes. As mentioned above, in the preferred embodiment, the multimode rake receiver 50 of the present invention is embodied as an ACE 100 or as one or more matrices 150 (with corresponding interconnection networks).

~~A significant departure from the prior art, the ACE 100 does not utilize~~ traditional (and typically separate) data, DMA, random access, configuration and instruction busses for signaling and other transmission between and among the reconfigurable matrices 150, the controller 120, and the memory 140, or for other input/output ("I/O") functionality. Rather, data, control (such as power and timing information) and configuration information are transmitted between and among these matrix 150 elements, utilizing the matrix interconnection network 110, which may be configured and reconfigured, to provide any given connection between and among the reconfigurable matrices 150, including those matrices 150 configured as the controller ~~120 and the memory 140, as discussed in greater detail below.~~

It should also be noted that once configured, the MIN 110 also and effectively functions as a memory, directly providing the interconnections for particular functions, until and unless it is reconfigured. In addition, such configuration and reconfiguration may occur in advance of the use of a particular function or operation, and/or may occur in real-time or at a slower rate, namely, in advance of, during or concurrently with the use of the particular function or operation. Such configuration and reconfiguration, moreover, may be occurring in a distributed fashion without disruption of function or operation, with computational elements in one location being configured while other computational elements (having been previously configured) are concurrently

performing their designated function. This configuration flexibility of the ACE 100 contrasts starkly with FPGA reconfiguration, both which generally occurs comparatively slowly, not in real-time or concurrently with use, and which must be completed in its entirety prior to any operation or other use.

~~The matrices 150 configured to function as memory 140 may be~~ implemented in any desired or preferred way, utilizing computational elements (discussed below) of fixed memory elements, and may be included within the ACE 100 or incorporated within another IC or portion of an IC (such as memory 61). In the preferred embodiment, the memory 140 is included within the ACE 100, and preferably is comprised of computational elements which are low power consumption random access memory (RAM), but also may be comprised of computational elements of any other form of memory, such as flash, DRAM, SRAM, MRAM, ROM, EPROM or E^2PROM. As mentioned, this memory functionality may also be distributed across multiple matrices 150, and may be temporally embedded, at any given time, as a particular MIN 110 configuration. In addition, in the preferred embodiment, the memory ~~140 preferably includes direct memory access (DMA) engines, not separately illustrated.~~

The controller 120 is preferably implemented, using matrices 150A and 150B configured as adaptive finite state machines, as a reduced instruction set ("RISC") processor, controller or other device or IC capable of performing the two types of functionality discussed below. (Alternatively, these functions may be implemented utilizing a conventional RISC or other processor, such as a processing element 65 of Figure 3.) This control functionality may also be distributed throughout one or more matrices 150 which perform other, additional functions as well. In addition, this control functionality may be included within and directly embodied as configuration information, without separate hardware controller functionality. The first control functionality, referred to as "kernel" control, is illustrated as kernel controller ("KARC") of matrix 150A, and the second control functionality, referred to as "matrix" control, is illustrated as matrix controller ("MARC") of matrix 150B. The kernel and matrix control functions of the controller 120 are explained in greater detail below, with reference to the configurability and reconfigurability of the various matrices 150, and with reference to the preferred form of combined data, configuration (and other control) information

referred to herein interchangeably as "silverware" or as a "silverware" module, such as a module 70.

~~The matrix interconnection network 110 of Figure 4, and its subset~~ interconnection networks separately illustrated in Figures 5 and 6 (Boolean interconnection network 210, data interconnection network 240, and interconnect 220), collectively and generally referred to herein as "interconnect", "interconnection(s)", "interconnection network(s)" or MIN, may be implemented generally as known in the art, such as utilizing field programmable gate array ("FPGA") interconnection networks or switching fabrics, albeit in a considerably more varied fashion. (As used herein, "field programmability" refers to the capability for post-fabrication adding or changing of actual IC functionality, as opposed to programming of existing IC structure or function (such as in a microprocessor or DSP). In the preferred embodiment, the various interconnection networks are implemented as described, for example, in U.S. Patent No. 5,218,240, U.S. Patent No. 5,336,950, U.S. Patent No. 5,245,227, and U.S. Patent No. 5,144,166, and also as discussed below and as illustrated with reference to Figures 8, 9 and 10. These various interconnection networks provide selectable (routable or switchable) connections between and among the controller 120, the memory 140, the various matrices 150, and the computational units 200 and computational elements 250 discussed below, providing the physical basis for the configuration and reconfiguration referred to herein, in response to and under the control of configuration signaling generally referred to herein as "configuration information" (and provided in modules 70). In addition, the various interconnection networks (110, 210, 240 and 220) provide selectable or switchable data, input, output, control and configuration paths, between and among the controller 120, the memory 140, the various matrices 150, and the computational units 200 and computational elements 250, in lieu of any form of traditional or separate input/output busses, data busses, DMA, RAM, configuration and ~~instruction busses.~~

It should be pointed out, however, that while any given switching or selecting operation of or within the various interconnection networks (110, 210, 240 and 220) may be implemented as known in the art, the design and layout of the various interconnection networks (110, 210, 240 and 220), in accordance with the present

invention, are new and novel, as discussed in greater detail below. For example, varying levels of interconnection are provided to correspond to the varying levels of the matrices 150, the computational units 200, and the computational elements 250, discussed below. At the matrix 150 level, in comparison with the prior art FPGA interconnect, the matrix interconnection network 110 is considerably more limited and less "rich", with lesser connection capability in a given area, to reduce capacitance and increase speed of operation. Within a particular matrix 150 or computational unit 200, however, the interconnection network (210, 220 and 240) may be considerably more dense and rich, to provide greater adaptation and reconfiguration capability within a narrow or close locality of reference.

The various matrices or nodes 150 are reconfigurable and heterogeneous, namely, in general, and depending upon the desired configuration: reconfigurable matrix 150A is generally different from reconfigurable matrices 150B through 150N; reconfigurable matrix 150B is generally different from reconfigurable matrices 150A and 150C through 150N; reconfigurable matrix 150C is generally different from reconfigurable matrices 150A, 150B and 150D through 150N, and so on. The various reconfigurable matrices 150 each generally contain a different or varied mix of adaptive and reconfigurable computational (or computation) units (200); the computational units 200, in turn, generally contain a different or varied mix of fixed, application specific computational elements (250), discussed in greater detail below with reference to Figures 4, 5 and 6, which may be adaptively connected, configured and reconfigured in various ways to perform varied functions, through the various interconnection networks. In addition to varied internal configurations and reconfigurations, the various matrices 150 may be connected, configured and reconfigured at a higher level, with respect to each of the other matrices 150, through the matrix interconnection network 110, also as discussed in greater detail below.

Several different, insightful and novel concepts are incorporated within the ACE 100 architecture of the present invention, and provide a useful explanatory basis for the real-time operation of the ACE 100 and its inherent advantages.

The first novel concepts of the present invention concern the adaptive and reconfigurable use of application specific, dedicated or fixed hardware units

(computational elements 250), and the selection of particular functions for acceleration, to be included within these application specific, dedicated or fixed hardware units (computational elements 250) within the computational units 200 (Figure 5) of the matrices 150, such as pluralities of multipliers, complex multipliers, and adders, each of which are designed for optimal execution of corresponding multiplication, complex multiplication, and addition functions. Given that the ACE 100 is to be optimized, in the preferred embodiment, for low power consumption, the functions for acceleration are selected based upon power consumption. For example, for a given application such as mobile communication, corresponding C (or C++) or other code may be analyzed for power consumption. Such empirical analysis may reveal, for example, that a small portion of such code, such as 10%, actually consumes 90% of the operating power when executed. In accordance with the present invention, on the basis of such power utilization, this small portion of code is selected for acceleration within certain types of the reconfigurable matrices 150, with the remaining code, for example, adapted to run within matrices 150 configured as controller 120. Additional code may also be selected for acceleration, resulting in an optimization of power consumption by the ACE 100, up to any potential trade-off resulting from design or operational complexity. In addition, as discussed with respect to Figure 5, other functionality, such as control code, may be accelerated within matrices 150 when configured as finite state machines. Through the varying levels of interconnect, corresponding algorithms are then implemented, at any given time, through the configuration and reconfiguration of fixed computational elements (250), namely, implemented within hardware which has been optimized and configured for efficiency, *i.e.*, a "machine" is configured in real-time which is optimized to perform the particular algorithm.

The next and perhaps most significant concept of the present invention, and a marked departure from the concepts and precepts of the prior art, is the concept of reconfigurable "heterogeneity" utilized to implement the various selected algorithms mentioned above. As indicated in the related application, prior art reconfigurability has relied exclusively on homogeneous FPGAs, in which identical blocks of logic gates are repeated as an array within a rich, programmable interconnect, with the interconnect subsequently configured to provide connections between and among the identical gates to

implement a particular function, albeit inefficiently and often with routing and combinatorial problems. In stark contrast, in accordance with the present invention, within computation units 200, different computational elements (250) are implemented directly as correspondingly different fixed (or dedicated) application specific hardware, such as dedicated multipliers, complex multipliers, and adders. Utilizing interconnect (210 and 220), these differing, heterogeneous computational elements (250) may then be adaptively configured, in advance, in real-time or perhaps at a slower rate, to perform the selected algorithm, such as the performance of discrete cosine transformations often utilized in mobile communications. As a consequence, in accordance with the present invention, different ("heterogeneous") computational elements (250) are configured and reconfigured, at any given time, to optimally perform a given algorithm or other function. In addition, for repetitive functions, a given instantiation or configuration of computational elements may also remain in place over time, *i.e.*, unchanged, throughout the course of such repetitive calculations. Such temporal stability of a given configuration may be indicated in a module 70, for example, through a loop field (discussed above), or simply left in place by not providing another (competing) configuration of the same computational elements.

The temporal nature of the ACE 100 architecture should also be noted. At any given instant of time, utilizing different levels of interconnect (110, 210, 240 and 220), a particular configuration may exist within the ACE 100 which has been optimized to perform a given function or implement a particular algorithm, such as to implement pilot signal searching for a CDMA operating mode in a mobile station 30 or 32. At another instant in time, the configuration may be changed, to interconnect other computational elements (250) or connect the same computational elements 250 differently, for the performance of another function or algorithm, such as multipath reception for a CDMA operating mode. Two important features arise from this temporal reconfigurability. First, as algorithms may change over time to, for example, implement a new technology standard, the ACE 100 may co-evolve and be reconfigured to implement the new algorithm. Second, because computational elements are interconnected at one instant in time, as an instantiation of a given algorithm, and then reconfigured at another instant in time for performance of another, different algorithm,

gate (or transistor) utilization is maximized, providing significantly better performance than the most efficient ASICs relative to their activity factors. This temporal reconfigurability also illustrates the memory functionality inherent in the MIN 110, as mentioned above.

This temporal reconfigurability of computational elements 250, for the performance of various different algorithms, also illustrates a conceptual distinction utilized herein between configuration and reconfiguration, on the one hand, and programming or reprogrammability, on the other hand. Typical programmability utilizes a pre-existing group or set of functions, which may be called in various orders, over time, to implement a particular algorithm. In contrast, configurability and reconfigurability, as used herein, includes the additional capability of adding or creating new functions which were previously unavailable or non-existent.

~~Next, the present invention also utilizes a tight coupling (or~~ interdigitation) of data and configuration (or other control) information, within a plurality of packets or within one, effectively continuous stream of information. This coupling or commingling of data and configuration information, referred to as "silverware" or as a "silverware" module, is illustrated in Figure 1. This coupling of data and configuration information into one information (or bit) stream, which may be continuous or divided into packets, helps to enable real-time reconfigurability of the ACE 100, without a need for the (often unused) multiple, overlaying networks of hardware interconnections of the prior art. For example, as an analogy, a particular, first configuration of computational elements at a particular, first period of time, as the hardware to execute a corresponding algorithm during or after that first period of time, may be viewed or conceptualized as a hardware analog of "calling" a subroutine in software which may perform the same algorithm. As a consequence, once the configuration of the computational elements has occurred (*i.e.,* is in place), as directed by (a first subset of) the configuration information, the data for use in the algorithm is immediately available as part of the silverware module. Referring to Figure 1, this is illustrated by "calling" various configurations (through references or flags in fields 80 and 81, for example, for configurations "A" and "B"), closely followed by providing the data for use in these configurations (fields 83 and 84 for configuration "A", fields 86 and 87 for configuration "B"). The same

computational elements may then be reconfigured for a second period of time, as directed by second configuration information (*i.e.,* a second subset of configuration information), for execution of a second, different algorithm, also utilizing immediately available data. The immediacy of the data, for use in the configured computational elements, provides a one or two clock cycle hardware analog to the multiple and separate software steps of determining a memory address and fetching stored data from the addressed registers. This has the further result of additional efficiency, as the configured computational elements may execute, in comparatively few clock cycles, an algorithm which may require orders of magnitude more clock cycles for execution if called as a subroutine in a ~~conventional microprocessor or digital signal processor ("DSP").~~

This use of silverware modules, such as module 70, as a commingling of data and configuration information, in conjunction with the reconfigurability of a plurality of heterogeneous and fixed computational elements 250 to form adaptive, different and heterogeneous computation units 200 and matrices 150, enables the ACE 100 architecture to have multiple and different modes of operation. For example, when included within a hand-held device, given a corresponding silverware module, the ACE 100 may have various and different operating modes as a cellular or other mobile telephone, a music player, a pager, a personal digital assistant, and other new or existing functionalities. In addition, these operating modes may change based upon the physical location of the device. For example, in accordance with the present invention, while configured for a first operating mode, using a first set of configuration information, as a CDMA mobile telephone for use in the United States, the ACE 100 may be reconfigured using a second set of configuration information for an operating mode as a GSM mobile telephone for use in Europe.

~~Referring again to Figure 4, the functions of the controller 120 (preferably~~ matrix (KARC) 150A and matrix (MARC) 150B, configured as finite state machines) may be explained with reference to a silverware module, namely, the tight coupling of data and configuration information within a single stream of information, with reference to multiple potential modes of operation, with reference to the reconfigurable matrices 150, and with reference to the reconfigurable computation units 200 and the computational elements 250 illustrated in Figure 5. As indicated above, through a

silverware module, the ACE 100 may be configured or reconfigured to perform a new or additional function, such as an upgrade to a new technology standard or the addition of an entirely new function, such as the addition of a music function to a mobile communication device. Such a silverware module may be stored in the matrices 150 of memory 140, or may be input from an external (wired or wireless) source through, for example, matrix interconnection network 110. In the preferred embodiment, one of the plurality of matrices 150 is configured to decrypt such a module and verify its validity, for security purposes. Next, prior to any configuration or reconfiguration of existing ACE 100 resources, the controller 120, through the matrix (KARC) 150A, checks and verifies that the configuration or reconfiguration may occur without adversely affecting any pre-existing functionality, such as whether the addition of music functionality would adversely affect pre-existing mobile communications functionality. In the preferred embodiment, the system requirements for such configuration or reconfiguration are included within the silverware module or configuration information, for use by the matrix (KARC) 150A in performing this evaluative function. If the configuration or reconfiguration may occur without such adverse affects, the silverware module is allowed to load into the matrices 150 (of memory 140), with the matrix (KARC) 150A setting up the DMA engines within the matrices 150C and 150D of the memory 140 (or other stand-alone DMA engines of a conventional memory). If the configuration or reconfiguration would or may have such adverse affects, the matrix (KARC) 150A does not allow the ~~new module to be incorporated within the ACE 100.~~

Continuing to refer to Figure 4, the matrix (MARC) 150B manages the scheduling of matrix 150 resources, clocking and the timing of any corresponding data, to synchronize any configuration or reconfiguration of the various computational elements 250 and computation units 200 with any corresponding input data and output data. In the preferred embodiment, timing or other clocking information is also included within a silverware module, to allow the matrix (MARC) 150B through the various interconnection networks to direct a reconfiguration of the various matrices 150 in time, and preferably just in time, for the reconfiguration to occur before corresponding data has appeared at any inputs of the various reconfigured computation units 200. In addition,

the matrix (MARC) 150B may also perform any residual processing which has not been accelerated within any of the various matrices 150.

This timing information may be embodied, for example, as the references or flags in fields 80, 81, 85, and 89 as illustrated in module 70 of Figure 1, to "call" the various configurations prior to the arrival of corresponding data (fields 84, 87, 91, 93, 95 and 97). In other circumstances, such as when configuration information has been provided to an ACE 100 in advance of and separately from user data, such as in mobile communications, this information may be injected or inserted into a user data stream for example, when transmitted or downloaded, to "call" appropriate configurations in advance of the reception of corresponding user data. In other circumstances, the matrix (MARC) 150B may itself insert these configuration references or flags, in real-time, into the data stream that is being processed by the various other matrices 150, to "call" and configure the appropriate computational elements 250. In addition, the matrix (MARC) 150B may also provide and insert the configuration and data routing information, for self-routing of the configuration information and the user data within the various matrices 150 (illustrated as fields 72, 74, 76, 78, 83, 86, 88, 90, 92, 94, and 96 in Figure 1), may provide and insert the power control fields (field 82) (to independently providing clocking (on or off) to any computational elements of the IC) and the other fields to create a module 70, such as fields 98 and 99 for information types and loop instructions. As a consequence, when an ACE 100 has not been provided with a module 70 directly, but has been provided with configuration information separately from user data, the matrix (MARC) 150B effectively creates such a module 70 for use in configuring the other matrices 150 to create the appropriate operating mode and use or operate upon the user data (incoming and outgoing).

As a consequence, the matrix (MARC) 150B may be viewed as a control unit which "calls" the configurations and reconfigurations of the matrices 150, computation units 200 and computational elements 250, in real-time, in synchronization or coordination with any corresponding data to be utilized by these various reconfigurable hardware units, and which performs any residual or other control processing. Other matrices 150 may also include this control functionality, with any

given matrix 150 capable of calling and controlling a configuration and reconfiguration of other matrices 150.

Figure 5 is a block diagram illustrating, in greater detail, a reconfigurable matrix 150 with a plurality of computation units 200 (illustrated as computation units 200A through 200N), and a plurality of computational elements 250 (illustrated as computational elements 250A through 250Z), and provides additional illustration of the preferred types of computational elements 250. As illustrated in Figure 5, any matrix 150 generally includes a matrix controller 230, a plurality of computation (or computational) units 200, and as logical or conceptual subsets or portions of the matrix interconnect network 110, a data interconnect network 240 and a Boolean interconnect network 210. As mentioned above, in the preferred embodiment, at increasing "depths" within the ACE 100 architecture, the interconnect networks become increasingly rich, for greater levels of adaptability and reconfiguration. The Boolean interconnect network 210, also as mentioned above, provides the reconfiguration and data interconnection capability between and among the various computation units 200, and is preferably small (*i.e.*, only a few bits wide), while the data interconnect network 240 provides the reconfiguration and data interconnection capability for data input and output between and among the various computation units 200, and is preferably comparatively large (*i.e.*, many bits wide). It should be noted, however, that while conceptually divided into reconfiguration and data capabilities, any given physical portion of the matrix interconnection network 110, at any given time, may be operating as either the Boolean interconnect network 210, the data interconnect network 240, the lowest level interconnect 220 (between and among the various computational elements 250), or other input, output, or connection functionality.

Continuing to refer to Figure 5, included within a computation unit 200 are a plurality of computational elements 250, illustrated as computational elements 250A through 250Z (individually and collectively referred to as computational elements 250), and additional interconnect 220. The interconnect 220 provides the reconfigurable interconnection capability and input/output paths between and among the various computational elements 250. As indicated above, each of the various computational elements 250 consist of dedicated, application specific hardware designed to perform a

given task or range of tasks, resulting in a plurality of different, fixed computational elements 250. Utilizing the interconnect 220, the fixed computational elements 250 may be reconfigurably connected together into adaptive and varied computational units 200, which also may be further reconfigured and interconnected, to execute an algorithm or other function, at any given time, utilizing the interconnect 220, the Boolean network 210, and the matrix interconnection network 110.

In the preferred embodiment, the various computational elements 250 are designed and grouped together, into the various adaptive and reconfigurable computation units 200 (as illustrated, for example, in Figures 6 through 10). In addition to computational elements 250 which are designed to execute a particular algorithm or function, such as multiplication, correlation, or addition, other types of computational elements 250 are also utilized in the preferred embodiment. As illustrated in Figure 5, computational elements 250A and 250B implement memory, to provide local memory elements for any given calculation or processing function (compared to the more "remote" memory 140). In addition, computational elements 250I, 250J, 250K and 250L are configured to implement finite state machines (using, for example, the computational elements illustrated in Figures 8, 9 and 10), to provide local processing capability (compared to the more "remote" matrix (MARC) 150B), especially suitable for complicated control processing.

With the various types of different computational elements 250 which may be available, depending upon the desired functionality of the ACE 100, the computation units 200 may be loosely categorized. A first category of computation units 200 includes computational elements 250 performing linear operations, such as multiplication, addition, finite impulse response filtering, and so on (as illustrated below, for example, with reference to Figure 7). A second category of computation units 200 includes computational elements 250 performing non-linear operations, such as discrete cosine transformation, trigonometric calculations, and complex multiplications. A third type of computation unit 200 implements a finite state machine, such as computation unit 200C as illustrated in Figure 5 and as illustrated in greater detail below with respect to Figures 8 through 10), particularly useful for complicated control sequences, dynamic scheduling, and input/output management, while a fourth type may implement memory and memory

management, such as computation unit 200A as illustrated in Figure 5. Lastly, a fifth type of computation unit 200 may be included to perform bit-level manipulation, such as for encryption, decryption, channel coding, Viterbi decoding, packet and protocol processing (such as Internet Protocol processing), and other types of processing and functions.

In the preferred embodiment, in addition to control from other matrices or nodes 150, a matrix controller 230 may also be included or distributed within any given matrix 150, also to provide greater locality of reference and control of any reconfiguration processes and any corresponding data manipulations. For example, once a reconfiguration of computational elements 250 has occurred within any given computation unit 200, the matrix controller 230 may direct that that particular instantiation (or configuration) remain intact for a certain period of time to, for example, continue repetitive data processing for a given application.

Figure 6 is a block diagram illustrating, in greater detail, an exemplary or representative computation unit 200 of a reconfigurable matrix 150 in accordance with the present invention. As illustrated in Figure 6, a computation unit 200 typically includes a plurality of diverse, heterogeneous and fixed computational elements 250, such as a plurality of memory computational elements 250A and 250B, and forming a computational unit ("CU") core 260, a plurality of algorithmic or finite state machine computational elements 250C through 250K. As discussed above, each computational element 250, of the plurality of diverse computational elements 250, is a fixed or dedicated, application specific circuit, designed and having a corresponding logic gate layout to perform a specific function or algorithm, such as addition or multiplication. In addition, the various memory computational elements 250A and 250B may be implemented with various bit depths, such as RAM (having significant depth), or as a register, having a depth of 1 or 2 bits.

Forming the conceptual data and Boolean interconnect networks 240 and 210, respectively, the exemplary computation unit 200 also includes a plurality of input multiplexers 280, a plurality of input lines (or wires) 281, and for the output of the CU core 260 (illustrated as line or wire 270), a plurality of output demultiplexers 285 and 290, and a plurality of output lines (or wires) 291. Through the input multiplexers 280,

an appropriate input line 281 may be selected for input use in data transformation and in the configuration and interconnection processes, and through the output demultiplexers 285 and 290, an output or multiple outputs may be placed on a selected output line 291, also for use in additional data transformation and in the configuration and interconnection processes.

Sub A10 ~~In the preferred embodiment, the selection of various input and output~~ lines 281 and 291, and the creation of various connections through the interconnect (210, 220 and 240), is under control of control bits 265 from the computational unit controller 255, as discussed below. Based upon these control bits 265, any of the various input enables 251, input selects 252, output selects 253, MUX selects 254, DEMUX enables 256, DEMUX selects 257, and DEMUX output selects 258, may be activated or ~~deactivated.~~

Sub A11 ~~The exemplary computation unit 200 includes a computation unit~~ controller 255 which provides control, through control bits 265, over what each computational element 250, interconnect (210, 220 and 240), and other elements (above) does with every clock cycle. Not separately illustrated, through the interconnect (210, 220 and 240), the various control bits 265 are distributed, as may be needed, to the various portions of the computation unit 200, such as the various input enables 251, input selects 252, output selects 253, MUX selects 254, DEMUX enables 256, DEMUX selects 257, and DEMUX output selects 258. The CU controller 295 also includes one or more lines 295 for reception of control (or configuration) information and transmission of ~~status information.~~

As mentioned above, the interconnect may include a conceptual division into a data interconnect network 240 and a Boolean interconnect network 210, of varying bit widths, as mentioned above. In general, the (wider) data interconnection network 240 is utilized for creating configurable and reconfigurable connections, for corresponding routing of data and configuration information. The (narrower) Boolean interconnect network 210, while also utilized for creating configurable and reconfigurable connections, is utilized for control of logic (or Boolean) decisions of data flow graphs (DFGs), generating decision nodes in such DFGs, and may also be used for data routing within such DFGs.

Figure 7 is a block diagram illustrating, in detail, an exemplary, preferred multi-function adaptive computational unit 500 having a plurality of different, fixed computational elements, in accordance with the present invention. When configured accordingly, the adaptive computation unit 500 performs a wide variety of functions discussed in the related application, such as finite impulse response filtering, fast Fourier transformation, and other functions such as discrete cosine transformation, useful for communication operating modes. As illustrated, this multi-function adaptive computational unit 500 includes capability for a plurality of configurations of a plurality of fixed computational elements, including input memory 520, data memory 525, registers 530 (illustrated as registers 530A through 530Q), multipliers 540 (illustrated as multipliers 540A through 540D), adder 545, first arithmetic logic unit (ALU) 550 (illustrated as ALU_1s 550A through 550D), second arithmetic logic unit (ALU) 555 (illustrated as ALU_2s 555A through 555D), and pipeline (length 1) register 560, with inputs 505, lines 515, outputs 570, and multiplexers (MUXes or MXes) 510 (illustrates as MUXes and MXes 510A through 510KK) forming an interconnection network (210, 220 and 240). The two different ALUs 550 and 555 are preferably utilized, for example, for parallel addition and subtraction operations, particularly useful for radix 2 operations in discrete cosine transformation.

Figure 8 is a block diagram illustrating, in detail, a preferred adaptive logic processor (ALP) computational unit 600 having a plurality of fixed computational elements, in accordance with the present invention. The ALP 600 is highly adaptable, and is preferably utilized for input/output configuration, finite state machine implementation, general field programmability, and bit manipulation. The fixed computational element of ALP 600 is a portion (650) of each of the plurality of adaptive core cells (CCs) 610 (Figure 9), as separately illustrated in Figure 10. An interconnection network (210, 220 and 240) is formed from various combinations and permutations of the pluralities of vertical inputs (VIs) 615, vertical repeaters (VRs) 620, vertical outputs (VOs) 625, horizontal repeaters (HRs) 630, horizontal terminators (HTs) 635, and horizontal controllers (HCs) 640.

~~Figure 9 is a block diagram illustrating, in greater detail, a preferred core~~ cell 610 of an adaptive logic processor computational unit 600 with a fixed computational

element 650, in accordance with the present invention. The fixed computational element is a 3-input —2 output function generator 550, separately illustrated in Figure 10. The preferred core cell 610 also includes control logic 655, control inputs 665, control outputs 670 (providing output interconnect), output 675, and inputs (with interconnect muxes) 660 (providing input interconnect).

Figure 10 is a block diagram illustrating, in greater detail, a preferred fixed computational element 650 of a core cell 610 of an adaptive logic processor computational unit 600, in accordance with the present invention. The fixed computational element 650 is comprised of a fixed layout of pluralities of exclusive NOR (XNOR) gates 680, NOR gates 685, NAND gates 690, and exclusive OR (XOR) gates 695, with three inputs 720 and two outputs 710. Configuration and interconnection is provided through MUX 705 and interconnect inputs 730.

As may be apparent from the discussion above, this use of a plurality of fixed, heterogeneous computational elements (250), which may be configured and reconfigured to form heterogeneous computation units (200), which further may be configured and reconfigured to form heterogeneous matrices 150, through the varying levels of interconnect (110, 210, 240 and 220), creates an entirely new class or category of integrated circuit, which may be referred to interchangeably as an adaptive computing architecture or adaptive computing engine. It should be noted that the adaptive computing architecture of the present invention cannot be adequately characterized, from a conceptual or from a nomenclature point of view, within the rubric or categories of FPGAs, ASICs or processors. For example, the non-FPGA character of the adaptive computing architecture is immediately apparent because the adaptive computing architecture does not comprise either an array of identical logical units, or more simply, a repeating array of any kind. Also for example, the non-ASIC character of the adaptive computing architecture is immediately apparent because the adaptive computing architecture is not application specific, but provides multiple modes of functionality and is reconfigurable, preferably in real-time. Continuing with the example, the non-processor character of the adaptive computing architecture is immediately apparent because the adaptive computing architecture becomes configured, to directly operate

upon data, rather than focusing upon executing instructions with data manipulation occurring as a byproduct.

Referring again to Figures 1 and 2, the various systems and methodology of the present invention may now be viewed in context of the ACE 100 architecture, based upon configuration and/or reconfiguration of fixed computational elements 250 in response to one or more sets of configuration information. Without the "something more" of configuration information, an ACE 100 is essentially or effectively an empty or "blank" device. Configuration information is necessary to generate the configurations creating one or more operating modes for the ACE 100, in order to provide a desired functionality and operate upon corresponding data, such as wireless communication, radio reception, or MP3 music playing.

Such configuration and reconfiguration may occur in a wide variety of ways. For example, an entire ACE 100 may be configured in advance of any particular use, such as pre-configured as a mobile communication device. In other embodiments, an ACE 100 may be configured to have an operating system, to power on (boot), and obtain and load other configurations for particular operating modes and functions, such as through a network 40. An ACE 100 may also be partially configured, with some matrices 150 configured and operating, while other matrices 150 are being configured for other functions.

Such an operating system in the ACE 100 may provide for a variety of automatic functions. For example, such an OS may provide for auto-routing, inserting routing fields and routing information, with configuration information, into data streams, to internally create a silverware module. Operating systems may also provide means to self-configure or self-modify, for example, using neural network and other self-learning technologies. Other operating system functions include authorization, security, hardware capability determinations, and other functions, as discussed below.

As mentioned above, such configuration information may be interleaved with data to form silverware (or a silverware module), such as executable module 70. In addition, such configuration information may also be separate from any data (effectively distributing a module 70 across time). For example, a first set of configuration information may be provided to an ACE 100 for a first operating mode, such as for

mobile communications. Data may be subsequently provided separately, such as voice data, during any given communication session. The various controller 120 functions of the ACE 100 then interleave the appropriate subsets of configuration information with corresponding data, routing, configuration references, loop instructions, and power control, to provide silverware modules to the matrices 150. As mentioned above, such controller functions may be distributed within the various matrices 150, or may be embedded within the configuration information itself.

Referring to Figure 2, an ACE 100 may obtain configuration information or entire silverware modules (70) from a plurality of sources. As illustrated in Figure 2, configuration information or one or more complete modules 70 may be provided to an ACE 100 through a download, from a server 54, WAN 42, LAN 41, or adjunct network entity 50, via a network 40 (with any applicable intervening switches 56 and 52 and base stations 25) or via a router 53, for example. The download may be either wireline (*e.g.* twisted pair, optical fiber, coaxial cable, hybrid fiber-coax) or wireless, such as through a transceiver of a base station 25 or satellite (not illustrated) or wireless LAN 43. The configuration information or one or more complete modules 70 may also be provided to an ACE 100 through other media, such as a flash memory 10, a silverware storage medium 15, a computer or other machine-readable medium 20, PCMCIA cards, PDA modules, or other memory cards, for example. This configuration information or one or more complete modules 70, in the preferred ACE 100 embodiment, is stored in memory 140, distributed memory within the various matrices 150, or in the system 60 (SOC) embodiment, may also be stored in memory 61. Configuration information may also simply be stored as an actual configuration of the matrices 150, with the MIN 110 effectively functioning as memory. The configuration information may also be transient, distributed and received in real-time for a particular application or for a singular use. Other equivalent provisioning and storage means will be apparent to those of skill in the art. (An ACE 100 receiving configuration information or one or more complete modules 70, through a download or other medium, is generally referred to herein as a "receiving" ACE.)

In addition, a need or request for such configuration information may also arise from a plurality of sources, including a system user, an element of infrastructure, an

ACE 100, another device including an ACE 100, or an independent device. For example, a system user may request a download of new configuration information to upgrade a device to a new standard, or may purchase a memory module (such as flash 10 or silverware storage medium 15) containing new configuration information or one or more complete modules 70 for playing additional, copyrighted MP3 music. Infrastructure elements may also initiate downloads of new configurations, either transmitted to an individual ACE 100 device (a single user, with a one-to-one (1:1) correspondence of provider and receiver) or broadcast to many ACE 100 devices (multiple users, with a one-to-many (1:many) correspondence of provider and receivers), to provide system upgrades, to adapt to new standards, or to provide other, real-time performance enhancements.

Another novel element of the present invention concerns a configuration or reconfiguration request generated by an ACE 100 itself (or another device including an ACE 100) providing, among other things, mechanisms for self-modification and self-configuration. For example, an ACE 100 (in a mobile station 30 or 32) typically having a first, CDMA configuration for use in the United States may be powered on in Europe; in the absence of standard CDMA signaling, the ACE 100 may request a wireless download of a second set of configuration information applicable to its current location, enabling the ACE 100 to have a GSM configuration for use in Europe.

As indicated above, configuration information is generally plural, consisting of a plurality of subsets of configuration information, such as first configuration information, second configuration information, through n^{th} configuration information. One "set" of configuration information may be considered to correspond to a particular operating mode of the ACE 100. For example, a first set of configuration information may provide a CDMA operating mode, while a second set of configuration information may provide a GSM operating mode.

Also as indicated above, for a given or selected higher-level operating mode of an ACE 100 (or, equivalently, for a given or selected set of configuration information), the various fixed, heterogeneous computational elements 250 are correspondingly configured and reconfigured for various lower-level or lower-order functional modes in response to the subsets of the configuration information, such as

configuration for discrete cosine transformation in response to first configuration information and reconfiguration for fast Fourier transformation in response to second configuration information.

The configuration information may also have different forms. In one embodiment, configuration information may include one or more discrete packets of binary information, which may be stored in memory 140, distributively stored within the matrices 150, or directly stored as a configuration of MIN 110. Configuration information may also be embodied in a continuous form, such as a continuous stream of binary or other information. As directed, configuration and other control bits from the configuration information are interdigitated with data to form silverware modules, for use in real-time within an ACE 100. In another embodiment, configuration information may be provided in real-time with corresponding data, in the form of a continuous stream (continuous for the duration of the selected function). For example, configuration information for a MP3 player may be provided in real-time in a silverware stream with the data bit file for the music to be played.

Two additional features are utilized to provide this capability for an ACE 100 to be safely and effectively configured and/or reconfigured in response to configuration information. First, a concept of "unit hardware", a parameter for or measurement of ACE 100 resources or capability, is utilized to gauge the capacity for a given ACE 100 to take on a new configuration and perform the new functionality, either in light of maintaining current configurations and functions and providing performance at sufficient or adequate levels, or in light of replacing current configurations and functions altogether. For example, a first generation ACE 100 may have sufficient resources, measured as unit hardware, to configure as a CDMA mobile station and simultaneously as a personal digital assistant. An attempt to load a new configuration, for example, for an MP3 player, may be inadvisable due to insufficient system resources, such that the new configuration would cause CDMA performance to degrade below acceptable levels. Conversely, a first generation ACE 100 initially configured as a PDA may have sufficient remaining resources to load the new configuration, as greater performance degradation may be allowable for these applications. Continuing with the example, a second or third generation ACE 100 may have sufficient computational element, interconnect and other

ACE 100 resources to support not only its currently existing configurations, but also such new configurations (with corresponding additional functionality), such as maintaining existing CDMA configurations while simultaneously having sufficient resources for additional GSM and MP3 configurations.

Related to this concept of unit hardware to measure reconfiguration capacity is the concept of multiple versions or libraries of configuration information or one or more complete modules 70 for the addition of new functionalities. Such multiple versions or libraries of configuration information or modules 70 are tailored to correspond to potentially differing capabilities of ACE 100 devices, particularly for application to the then current ACE architectures compared to legacy architectures. Such forward "binary compatibility" will allow a module 70, designed for a current ACE 100, to operate on any newer, future ACE. For example, a suite of different sets of configuration information may be developed to provide a particular operating mode, with differences pertaining to matters such as performance quality and the number and types of features. Each of the various sets or versions of the configuration information are generated to have system requirements corresponding to the available and varying levels of ACE 100 reconfiguration capacity. Such libraries of configuration information, having requirements levels corresponding to levels of "unit hardware", may be generated in advance of a requested download or other provision, or may be generated as needed, on a real-time basis, tailored to the particular configuration capacity of the receiving ACE 100. For example, corresponding, tailored configuration information downloads may be determined in real-time, based upon a negotiation or interactivity between the ACE 100 and the configuration provider, generating and providing configuration information suitable for a negotiated or predetermined level of performance for a given operating mode.

Also for example, configuration information for a particular operating mode may be available only with one version having predetermined system requirements. In that event, if the particular ACE 100 does not have the corresponding capacity to meet those requirements, the ACE 100 itself may reject or decline such a potential download.

As a consequence, prior to a configuration (and/or reconfiguration) of a particular ACE architecture for a particular operating mode, the capabilities of that ACE

100 are determined, to avoid a download or reception of a configuration which potentially may alter or harm pre-existing operating modes or other functionalities of the device, or to provide a more suitable download tailored for the capabilities of the particular ACE 100.

The nature of the malleable ACE 100 architecture, with different physical connections created or removed in response to configuration information, renders security for configuration and reconfiguration of paramount importance. Given that such configurations are capable of altering the operating mode of the ACE architecture, in the preferred method, system and apparatus embodiments, authorization and security measures are implemented to avoid potentially destructive or harmful configurations, such as due to viruses or other unwanted, rogue configuration information. In the preferred module 70 embodiment, such security information is included within the header field 71.

Several levels of security may be implemented to control the configurability and reconfigurability of an ACE 100. A first level of security is implemented at a level of authorization to request or receive configuration information. For example, an ACE 100 may have a unique identifier or digital signature transmitted to a server 54 during a "handshake" or other initial exchange of information (such as unit hardware information) prior to a download of configuration information. The server 54 may access a database of authorized recipients, and if the particular ACE 100 is included, the server 54 will authorize the download. Such authorization measures are important for the protection of intellectual property, such as copyrighted material, and other information which may be confidential or otherwise restricted. Another level of security may be implemented to protect against the possible download of rogue, virus or corrupted configuration information, utilizing various encryption and decryption technologies, for example.

Various forms of monitoring, tracking and other record keeping are also utilized for determining and accounting for the various configuration and content usage possibilities, and may involve numerous different network entities. For example, a particular download of a module 70 or other configuration information may be generated from more than one network entity, with one transaction for a particular download of a

module 70 or other configuration information also distributed across more than one network entity. Continuing with the example, a request for a download of a module 70 (or other configuration information or silverware) may be received at a base station 25 of a wireless service provider "A". To fulfill the request, the wireless service provider "A" determines the authorization status of the requesting ACE 100 and when authorized, forwards the request to another provider, such as content provider "B", which provides requested data, such as a music bit file, using a content server 54. Also in response to the request from provider "A", a set of MP3 configuration information is simultaneously provided by configuration provider "C", using a second, different server 54 under its control, such as a configuration information server. The content (data) and configuration information are provided to silverware module provider "D", who in turn interleaves the data and configuration to form a silverware module 70, using a first adjunct network entity 50 having a silverware module application node 51. Next, an encryption provider "E" encrypts the silverware module, using a second adjunct network entity 50 having an encryption application node 51, providing the encrypted silverware module to the service provider "A" for transmission to the requesting ACE 100. Corresponding accounting and other records may be generated for each such distributed transaction, with corresponding distributions of royalties, use and license fees. Content usage may also be tracked by, for example, a content server.

The generation and provision of configuration information may also be distributed across time, in addition to distributed across space, with the various functions referred to above performed during different intervals of time. For example, one or more versions or sets of configuration information may be generated and stored during a first predetermined period of time, such as in advance of any particular use. Subsequently, such a set of configuration information may be provided during a second predetermined period of time, such as following a security and financial authorization process.

In summary, the present invention provides a method of configuration and operation or an adaptive and reconfigurable circuit, preferably utilizing an executable module comprised of a plurality of information sequences. A first information sequence (or field) provides configuration control, which may be either configuration information or a reference (such as a flag or other designation) to corresponding configuration

information cached or stored in memory. A second information sequence provides operand data for use by configured computational elements. A third information sequence provides routing control, to direct the other information sequences to their appropriate locations within the matrix environment of the ACE integrated circuitry. Also in the preferred embodiment a fourth information sequence is utilized to provide power control, to clock on or off various computational elements, and a fifth information sequence may be utilized for loop or iteration control.

Also in summary, one of the preferred system embodiments provides, first, means for routing configuration information to a plurality of computational elements; second, means for configuring and reconfiguring a plurality of computational elements to form a plurality of configured computational elements for the performance of a plurality of selected functions; third, means for providing operand data to the plurality of configured computational elements; and fourth, means for controlling configuration timing to precede a receipt of corresponding operand data.

Another preferred system embodiment provides, first, means for spatially configuring and reconfiguring a plurality of computational elements to form a first plurality of configured computational elements for the performance of a first plurality of selected functions; second, means for temporally configuring and reconfiguring the plurality of computational elements to form a second plurality of configured computational elements for the performance of a second plurality of selected functions; third, means for providing data to the first and second pluralities of configured computational elements; and fourth, means for coordinating the spatial and temporal configurations of the plurality of computational elements with the provision of the data to the first and second pluralities of configured computational elements.

Numerous advantages of the various embodiments of the present invention are readily apparent. The present invention provides an apparatus, method and system for configuration and operation of adaptive integrated circuitry, to provide one or more operating modes or other functionality of ACE circuitry and other devices incorporating ACE technology. The apparatus, method and systems of the invention combine silverware modules or other configuration information with an ACE circuit (or ACE IC), for the provision of a selected operating mode. In addition, the various embodiments of

the present invention provide coordination of configuration with data reception and provide independent control of power usage for different portions of the IC.

Yet additional advantages of the present invention may be further apparent to those of skill in the art. The ACE 100 architecture of the present invention effectively and efficiently combines and maximizes the various advantages of processors, ASICs and FPGAs, while minimizing potential disadvantages. The ACE 100 includes the concepts or ideals of the programming flexibility of a processor, the post-fabrication flexibility of FPGAs, and the high speed and high utilization factors of an ASIC, with additional features of low power consumption and low cost. The ACE 100 is readily reconfigurable, in real-time, and is capable of having corresponding, multiple modes of operation. In addition, through the selection of particular functions for reconfigurable acceleration, the ACE 100 minimizes power consumption and is suitable for low power applications, such as for use in hand-held and other battery-powered devices.

From the foregoing, it will be observed that numerous variations and modifications may be effected without departing from the spirit and scope of the novel concept of the invention. It is to be understood that no limitation with respect to the specific methods and apparatus illustrated herein is intended or should be inferred. It is, of course, intended to cover by the appended claims all such modifications as fall within the scope of the claims.

It is claimed: